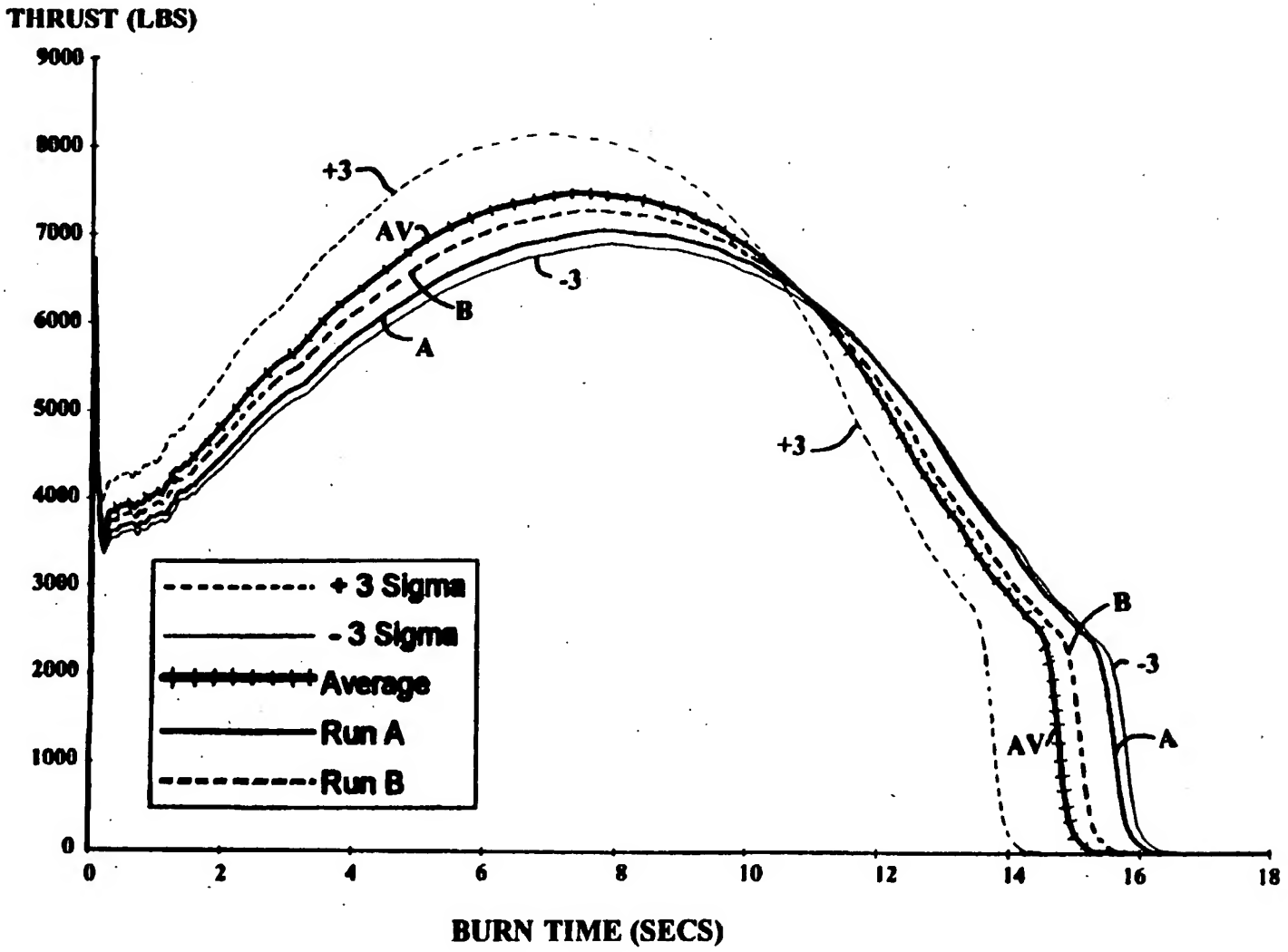
THRUST (LBS)
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
0
2
4
6
8
10
12
14
16
18
BURN TIME (SECS)
+3
AV
B
A
-3
+3
B
-3
A
AV
+ 3 Sigma
- 3 Sigma
Average
Run A
Run B

FIG. 1

```
Private Sub ClearOutValues_Click()
  AtFOut.Text = ""
  OutDensity.Text = ""
  PwtOut.Text = ""
  OutWebDepth.Text = ""
  OutN = ""
  OutBurnRate.Text = ""

End Sub
```

```
Private Sub ConvertButton_Click()

  Dim BSModifier As Double
  Dim ti As Single
  Dim pmax As Single, fmax As Single
  Dim pti As Double

  '   Set blank input or output variables
  If FillBlankValues() = False Then
    Exit Sub
  End If

  '   check if input data loaded
  If InLoaded = False Then
    MsgBox "Your need to load a data file."
    Exit Sub
  End If

  ' Turn Hourglass on
  MainModel.MousePointer = vbHourglass

  '   Calculate throat erosion profile to match initial and final throat values
  If ThrustFromAt = 0 Then
    Call SetErosion
  End If

  ' Adjust reference burn rate to match theoretical web burned
  Call SetBurnRate

  ' Adjust C* until propellant weight burned matches given weight
  Call SetCStar

  ' Plot input burn back profile

  DataCutFrame.Visible = False
  StatusLabel.Visible = True
  StatusText.Caption = "Plotting Burnback Profile..."
  StatusText.Visible = True
```

FIG. 2A

```
ProgressBar1.Value = 0
ProgressBar1.Visible = True
WebAtMaxPress.Caption = ""
Refresh

MainModel.ThrustGraph.ScaleHeight = MaxThrust * 1.4
MainModel.ThrustGraph.ScaleWidth = MaxTime * 1.4
MainModel.ThrustGraph.ScaleTop = MaxThrust * 0.1
MainModel.ThrustGraph.ScaleLeft = -MaxTime * 0.1
MainModel.BBPGraph.ScaleHeight = MaxBS * 1.4
MainModel.BBPGraph.ScaleWidth = MaxWeb * 1.4
MainModel.BBPGraph.ScaleTop = MaxBS * 0.1
MainModel.BBPGraph.ScaleLeft = -MaxWeb * 0.1

BBPGraph.Line (0, MaxBS * 1.3)-(0, MaxBS * 0.1)
BBPGraph.Line (0, MaxBS * 1.3)-(MaxWeb * 1.3, MaxBS * 1.3)
For i = 2 To InPoints
  If i / Int(InPoints / 10) = Int(i / Int(InPoints / 10)) Then
    ProgressBar1.Value = i / InPoints * 100
    Refresh
  End If
  BBPGraph.Line (WBurnIn(i - 1), MaxBS * 1.3 - BSurfIn(i - 1))-(WBurnIn(i), MaxBS
  *  1.3 - BSurfIn(i)), RED
Next i
BBPGraph.Line (WBurnIn(i - 1), MaxBS * 1.3 - BSurfIn(i - 1))-(WBurnIn(i - 1),
MaxBS * 1.3), RED

'    StatusLabel.Visible = False
'    StatusText.Visible = False
'    ProgressBar1.Visible = False
'    DataCutFrame.Visible = True
'    Refresh

' Write burn back file to disk
If BBPCheck.Value = Checked Then
  fnum = FreeFile
  BbpFileSpec = BBPFileName.Text
  On Error Resume Next
  Open BbpFileSpec For Output As #fnum
  If Err Then
    MsgBox Error$
    Exit Sub
  End If
  Print #fnum, "Source:" & Chr$(9) & InFileSpec
  Print #fnum, "Temp:" & Chr$(9) & InTemp.Text
  Print #fnum, "Initial Throat:" & Chr$(9) & At0.Text
  Print #fnum, "Final Throat:" & Chr$(9) & AtF.Text
  Print #fnum, "Prop Wt:" & Chr$(9) & Pwt.Text
  Print #fnum, "Burn Rate @ 1500psi:" & Chr$(9) & InBurnRate.Text
  Print #fnum, "C*:" & Chr$(9) & CStar.Text
  Print #fnum, ""
  Print #fnum, "Web Burned" & Chr$(9) & "Burn Surface" & Chr$(9) & "Throat Area" &
  Chr$(9) & "Cf" & Chr$(9)
  Print #fnum, ""

  For i = 1 To InPoints
    Print #fnum, WBurnIn(i) & Chr$(9) & BSurfIn(i) & Chr$(9) & AtIn(i) & Chr$(9)
    &   cfIn(i)
  Next i

  Close fnum
End If
```

FIG. 2B

```
' Generate converted file
If ProgramOptions(1) = True Then
   BSModifier = GenerateOutput()

   ' Plot Output thrust trace & burn back profile & write output file

   DataCutFrame.Visible = False
   StatusLabel.Visible = True
   StatusText.Caption = "Plotting Output data..."
   StatusText.Visible = True
   ProgressBar1.Value = 0
   ProgressBar1.Visible = True
   Refresh

   If OutputCheck.Value = Checked Then
      fnum = FreeFile
      OutFileSpec = OutputFileName.Text
      On Error Resume Next
      Open OutFileSpec For Output As #fnum
      If Err Then
         MsgBox Error$, , "Error Opening Output File"
         Exit Sub
      End If
      Print #fnum, "Source:" & Chr$(9) & InFileSpec
      Print #fnum, "From Temp:" & Chr$(9) & InTemp.Text
      Print #fnum, "To Temp:" & Chr$(9) & OutTemp.Text
      Print #fnum, "Initial Throat:" & Chr$(9) & At0.Text
      Print #fnum, "Final Throat:" & Chr$(9) & AtF.Text
      Print #fnum, "Prop Wt:" & Chr$(9) & Pwt.Text
      Print #fnum, "Burn Rate @ 1500psi & Input Temp:" & Chr$(9) & InBurnRate.Text
      Print #fnum, "Burn Rate @ 1500psi & Output Temp:" & Chr$(9) & OutBurnRate.
      Text
      Print #fnum, "C*:" & Chr$(9) & CStar.Text
      Print #fnum, "BSModifier" & Chr$(9) & BSModifier
      Print #fnum, ""
      Print #fnum, "Time" & Chr$(9) & "Pressure" & Chr$(9) & "Thrust" & Chr$(9) &
      "Throat Area" & Chr$(9) & "Cf" & Chr$(9) & "Web Burned" & Chr$(9) & "Burn
      Surface"
      Print #fnum, ""
   End If

   ThrustGraph.Cls
   BBPGraph.Cls

   BBPGraph.Line (0, MaxBS * 1.3)-(0, MaxBS * 0.1)
   BBPGraph.Line (0, MaxBS * 1.3)-(MaxWeb * 1.3, MaxBS * 1.3)
   ThrustGraph.Line (0, MaxThrust * 1.3)-(0, MaxThrust * 0.1)
   ThrustGraph.Line (0, MaxThrust * 1.3)-(MaxTime * 1.3, MaxThrust * 1.3)

   For i = 2 To InPoints
      If i / Int(InPoints / 10) = Int(i / Int(InPoints / 10)) Then
         ProgressBar1.Value = i / InPoints * 100
         Refresh
      End If

      ti = ti + (ThrustOut(i) + ThrustOut(i - 1)) / 2 * (TimeOut(i) - TimeOut(i -
      1))
      pti = pti + (PressOut(i) + PressOut(i - 1)) / 2 * (TimeOut(i) - TimeOut(i -
      1))
      If PressOut(i) > pmax Then
         pmax = PressOut(i)
         WebAtPmax = WBurnOut(i)
```

FIG. 2C

```
    End If
    If ThrustOut(i) > fmax Then
      fmax = ThrustOut(i)
    End If
    If bsmax < BSurfOut(i) Then
      bsmax = BSurfOut(i)
    End If

    ThrustGraph.Line (TimeIn(i - 1), MaxThrust * 1.3 - ThrustIn(i - 1))-(TimeIn(i
    ), MaxThrust * 1.3 - ThrustIn(i)), RED
    ThrustGraph.Line (TimeOut(i - 1), MaxThrust * 1.3 - ThrustOut(i - 1))-
    (TimeOut(i), MaxThrust * 1.3 - ThrustOut(i)), BLUE
    BBPGraph.Line (WBurnIn(i - 1), MaxBS * 1.3 - BSurfIn(i - 1))-(WBurnIn(i),
    MaxBS * 1.3 - BSurfIn(i)), RED
    BBPGraph.Line (WBurnOut(i - 1), MaxBS * 1.3 - BSurfOut(i - 1))-(WBurnOut(i),
    MaxBS * 1.3 - BSurfOut(i)), BLUE

    If OutputCheck.Value = Checked Then
      Print #fnum, TimeOut(i) & Chr$(9) & PressOut(i) & Chr$(9) & ThrustOut(i) &
      Chr$(9) & AtOut(i) & Chr$(9) & CfOut(i) & Chr$(9) & WBurnOut(i) & Chr$(9) &
      BSurfOut(i)
    End If

  Next i

  If OutputCheck.Value = Checked Then
    Close fnum
  End If

  ThrustGraph.Line (TimeIn(i - 1), MaxThrust * 1.3 - ThrustIn(i - 1))-(TimeIn(i -
  1 ), MaxThrust * 1.3 - 0), RED
  ThrustGraph.Line (TimeOut(i - 1), MaxThrust * 1.3 - ThrustOut(i - 1))-(TimeOut(i
  - 1), MaxThrust * 1.3 - 0), BLUE
  BBPGraph.Line (WBurnIn(i - 1), MaxBS * 1.3 - BSurfIn(i - 1))-(WBurnIn(i - 1),
  MaxBS * 1.3 - 0), RED
  BBPGraph.Line (WBurnOut(i - 1), MaxBS * 1.3 - BSurfOut(i - 1))-(WBurnOut(i - 1),
  MaxBS * 1.3 - 0), BLUE

End If

BurnTimeOut.Text = Format(TimeOut(i - 1), "0.000")
TIOut.Text = Format(ti, "0")
PTIOut.Text = Format(pti, "0")
PmaxOut.Text = Format(pmax, "0")
FmaxOut.Text = Format(fmax, "0")
WebAtMaxPress.Caption = Format(WebAtPmax, "0.00")

StatusLabel.Visible = False
StatusText.Visible = False
ProgressBar1.Visible = False
DataCutFrame.Visible = True
Refresh

' Turn Hourglass off
MainModel.MousePointer = vbDefault

End Sub
```

```
Private Function FillBlankValues() As Integer

  FillBlankValues = False
```

FIG. 2D

```
  If InTemp.Text = "" Or OutTemp.Text = "" Then
    MsgBox "You must enter both a Firing Temperature and an Output temperature."
    OutTemp.SetFocus
    Exit Function
  End If

-If At0.Text = "" Then
    MsgBox "You must enter an initial throat diameter."
    At0.SetFocus
    Exit Function
  End If
  If AtF.Text = "" Then
    MsgBox "You must enter a final throat diameter."
    AtF.SetFocus
    Exit Function
-End If
  If Pwt.Text = "" Then
    MsgBox "You must enter a propellant weight."
    Pwt.SetFocus
    Exit Function
-End If
  If InDensity.Text = "" Then
    InDensity.Text = Format(0.065, ".0000")   'eq(10)
-End If
-If InWebDepth.Text = "" Then
    MsgBox "You must enter a web thickness."
    InWebDepth.SetFocus
    Exit Function
  End If
-If InN.Text = "" Then
    MsgBox "You must enter a pressure exponent."
    InN.SetFocus
    Exit Function
-End If
  If SigmaP.Text = "" Then
    MsgBox "You must enter an temperature sensitivity coefficient."
    SigmaP.SetFocus
    Exit Function
-End If
  InBurnRate.Text = Format(0.0006 * Val(InTemp.Text) + 0.52, ".000")

  If PwtOut.Text = "" Then
    PwtOut.Text = Pwt.Text
-End If
-If OutDensity.Text = "" Then
    OutDensity.Text = InDensity.Text 'eq(10)
-End If
-If OutWebDepth.Text = "" Then
    OutWebDepth.Text = InWebDepth.Text
  End If
-If OutN.Text = "" Then
    OutN.Text = InN.Text
-End If
-If AtFOut.Text = "" Then
    AtFOut.Text = AtF.Text
-End If

  FillBlankValues = True

End Function
```

```
Private Sub Form_Load()
```

FIG. 2E

```
Private Sub SetErosion()
  'This subroutine varies the coefficient a in the equation E=aP^b
  'until the final throat calculated matches the given throat.
  'E is the incremental radial throat erosion and P is the incremental
  'chamber pressure.  The coefficent b is fixed at .924.  This routine build the
  throat area area that becomes
  'part of the burn back file

  Dim ThisValue As Double, Goal As Double
  Dim Step As Double, Direction As Integer, Accuracy As Double

  Dim a As Double, b As Double, c As Double, d As Double, Trial As Integer

  DataCutFrame.Visible = False
  StatusLabel.Visible = True
  StatusText.Caption = "Calculating Erosion Coefficients..."
  StatusText.Visible = True
  ProgressBar1.Value = 0
  ProgressBar1.Visible = True
  Refresh

  'Calculate erosion coefficients

  a = 0.00001477
  b = 0.924
  Goal = Val(AtF.Text)
  Accuracy = 0.00000001
  Direction = 0        'first time thru step will be halved
  Step = 0.00001       'before it is applied
  Trial = 0

  Do
    Trial = Trial + 1
    ThisValue = Val(At0.Text)
    EarlyEnd = InPoints
    For i = 1 To InPoints
      If i / Int(InPoints / 10) = Int(i / Int(InPoints / 10)) Then
        StatusText.Caption = "Calculating Erosion Coefficients... Trial #" & Trial
        ProgressBar1.Value = i / InPoints * 100
        Refresh
      End If
      ThisValue = ThisValue + (a * PressIn(i) ^ b) * (TimeIn(i) - TimeIn(i - 1)) *
      2     'eq(2)
      AtIn(i) = ThisValue ^ 2 / 4 * PI                    'eq(3)
```

FIG. 3A

```
    If i > InPoints / 2 And PressIn(i) < EOFiringPress.Text Then
      EarlyEnd = i
      Exit For
    End If

  Next i
  InPoints = EarlyEnd
  If Sgn(Goal - ThisValue) <> Sgn(Direction) Then
    Step = Step / 2
    Direction = Sgn(Goal - ThisValue)
  End If
  If Sgn(Goal - ThisValue) > 0 Then
    a = a + Step
  Else
    a = a - Step
  End If

Loop Until Abs(Goal - ThisValue) <= Accuracy

  StatusLabel.Visible = False
  StatusText.Visible = False
  ProgressBar1.Visible = False
  DataCutFrame.Visible = True
  Refresh

End Sub

Private Sub ThrustCoefFactor_MouseDown(Button As Integer, Shift As Integer, X
As Single, Y As Single)
  CfFactorPopup.Visible = True

End Sub

Private Sub ThrustCoefFactor_MouseUp(Button As Integer, Shift As Integer, X
As Single, Y As Single)
  CfFactorPopup.Visible = False

End Sub
```

FIG. 3B

```
Private Sub SetBurnRate()
  'This subroutine varies the reference burn rate at the input temperature
  'until the web burned at end of firing calculated from the input pressure
  'data matches the theoretical (or user entered) web. This subroutine
  'builds the web burned array, which becomes part of the burn back file

  Dim ThisValue As Double, Goal As Double
  Dim Step As Double, Direction As Integer, Accuracy As Double
```

FIG. 4A

```
Dim a As Double, b As Double, c As Double, d As Double

Dim rref As Double, brate As Double, trefin As Single, trefout As Single
Dim nexp As Single, PrevBr As Double, Trial As Integer

DataCutFrame.Visible = False
StatusLabel.Visible = True
StatusText.Caption = "Calculating Reference Burn Rate..."
StatusText.Visible = True
ProgressBar1.Value = 0
ProgressBar1.Visible = True

Goal = Val(InWebDepth.Text)
Accuracy = 0.001
Direction = 0        'first time thru step will be halved
Step = 0.1           'before it is applied
rref = Val(InBurnRate.Text)
trefin = Val(InTemp.Text)
nexp = Val(InN.Text)
PrevBr = Val(InBurnRate.Text)
Trial = 0

Do
  Trial = Trial + 1
  ThisValue = 0
  For i = 1 To InPoints
    If i / Int(InPoints / 10) = Int(i / Int(InPoints / 10)) Then
      StatusText.Caption = "Calculating Reference Burn Rate... Trial #" & Trial
      ProgressBar1.Value = i / InPoints * 100
      Refresh
    End If
    brate = rref * (PressIn(i) / PREF) ^ nexp    'eq 4
    ThisValue = ThisValue + (PrevBr + brate) / 2 * (TimeIn(i) - TimeIn(i - 1))
    'eq 5
    WBurnIn(i) = ThisValue
    PrevBr = brate
  Next i
  If Sgn(Goal - ThisValue) <> Sgn(Direction) Then
    Step = Step / 2
    Direction = Sgn(Goal - ThisValue)
  End If
  If Sgn(Goal - ThisValue) > 0 Then
    rref = rref + Step
  Else
    rref = rref - Step
  End If

Loop Until Abs(Goal - ThisValue) <= Accuracy

StatusLabel.Visible = False
StatusText.Visible = False
ProgressBar1.Visible = False
DataCutFrame.Visible = True
Refresh

InBurnRate.Text = rref
OutBurnRate.Text = Val(InBurnRate.Text) * Exp((Val(SigmaP.Text) * (Val(OutTemp.
Text) - Val(InTemp.Text))))

End Sub
```

FIG. 4B

```
Privat  Sub SetCStar()
  'This function varies the value of C* until the propellant weight burned
  'at the end of firing matches the given propellant weight.  This routine
  'builds the burn surface and thrust coefficient arrays that become part
  'of the burn back file.

  Dim ThisValue As Double, Goal As Double
  Dim Step As Double, Direction As Integer, Accuracy As Double

  Dim a As Double, b As Double, c As Double, d As Double

  Dim rref As Double, brate As Double, trefin As Single, trefout As Single
  Dim nexp As Single, mpdot As Double, PrevMpdot As Double, CS As Double
  Dim rho As Double, Trial As Integer
  Dim bsmax As Double

  DataCutFrame.Visible = False
  StatusLabel.Visible = True
  StatusText.Caption = "Matching Propellant Weight..."
  StatusText.Visible = True
  ProgressBar1.Value = 0
  ProgressBar1.Visible = True

  Goal = Val(Pwt.Text)
  Accuracy = 0.01
  Direction = 0        'first time thru step will be halved
  Step = 1000          'before it is applied
  rho = Val(InDensity.Text)
  rref = Val(InBurnRate.Text)
  trefin = Val(InTemp.Text)
  nexp = Val(InN.Text)
  CStar.Text = 60000 / 12
  CS = Val(CStar.Text) * 12
  Trial = 0

 Do
    Trial = Trial + 1
    ThisValue = 0
    bsmax = 0
   For i = 1 To InPoints
    If i / Int(InPoints / 10) = Int(i / Int(InPoints / 10)) Then
      StatusText.Caption = "Matching Propellant Weight... " & Chr$(13) & "Trial
      #" & Trial
      ProgressBar1.Value = i / InPoints * 100
      Refresh
    End If
     mpdot = PressIn(i) * AtIn(i) * G / CS     'eq 6
     brate = rref * (PressIn(i) / PREF) ^ nexp    'eq 4
     BSurfIn(i) = mpdot / (brate * rho)           'eq 4
    If bsmax < BSurfIn(i) Then
      bsmax = BSurfIn(i)
    End If
     ThisValue = ThisValue + (mpdot + PrevMpdot) / 2 * (TimeIn(i) - TimeIn(i - 1))
     'eq 8)
     PrevMpdot = mpdot
     cfIn(i) = ThrustIn(i) / (PressIn(i) * AtIn(i))   'eq 9
   Next i
   If Sgn(Goal - ThisValue) <> Sgn(Direction) Then
     Step = Step / 2
     Direction = Sgn(Goal - ThisValue)
   End If
   If Sgn(Goal - ThisValue) > 0 Then
```

FIG. 5A

```
    CS = CS - Step
  Else
    CS = CS + Step
  End If

Loop Until Abs(Goal - ThisValue) <= Accuracy

StatusLabel.Visible = False
StatusText.Visible = False
ProgressBar1.Visible = False
DataCutFrame.Visible = True
Refresh

CStar.Text = Format(CS / 12, "0")
MaxBS = bsmax * 1.2

End Sub
```

FIG. 5B

```
Private Sub SetErosion()
  'This subroutine varies the coefficient a in the equation E=aP^b
  'until the final throat calculated matches the given throat.
  'E is the incremental radial throat erosion and P is the incremental
  'chamber pressure.  The coefficent b is fixed at .924.  This routine build the
  throat area area that becomes
  'part of the burn back file

  Dim ThisValue As Double, Goal As Double
  Dim Step As Double, Direction As Integer, Accuracy As Double

  Dim a As Double, b As Double, c As Double, d As Double, Trial As Integer

  DataCutFrame.Visible = False
  StatusLabel.Visible = True
  StatusText.Caption = "Calculating Erosion Coefficients..."
  StatusText.Visible = True
  ProgressBar1.Value = 0
  ProgressBar1.Visible = True
  Refresh

  'Calculate erosion coefficients

  a = 0.00001477
  b = 0.924
  Goal = Val(AtF.Text)
  Accuracy = 0.00000001
  Direction = 0        'first time thru step will be halved
  Step = 0.00001       'before it is applied
  Trial = 0

  Do
    Trial = Trial + 1
    ThisValue = Val(At0.Text)
    EarlyEnd = InPoints
    For i = 1 To InPoints
      If i / Int(InPoints / 10) = Int(i / Int(InPoints / 10)) Then
        StatusText.Caption = "Calculating Erosion Coefficients... Trial #" & Trial
        ProgressBar1.Value = i / InPoints * 100
        Refresh
      End If
      ThisValue = ThisValue + (a * PressIn(i) ^ b) * (TimeIn(i) - TimeIn(i - 1)) *
      2    'eq(2)
      AtIn(i) = ThisValue ^ 2 / 4 * PI                    'eq(3)
```

```
Private Function GenerateOutput()
  'This function uses the burn surface profile, cf profile and throat area
  'profile generated from the input data to generate a prediction at the
  'output temperature.  The model uses an iterative process that varies a
  'burn surface modifier until the calculated propellant weight burned matches
  'the actual propellant weight.

  Dim ThisValue As Double, Goal As Double
  Dim Step As Double, Direction As Integer, Accuracy As Double

  Dim a As Double, b As Double, c As Double, d As Double

  Dim rref As Double, brate As Double, trefin As Single, trefout As Single
  Dim nexp As Single, mpdot As Double, PrevMpdot As Double, CS As Double
  Dim rho As Double, Trial As Integer
  Dim BSFactor As Double, BSModifier As Double, PercentWeb As Single

  DataCutFrame.Visible = False
  StatusLabel.Visible = True
  StatusText.Caption = "Generating Converted File..."
  StatusText.Visible = True
  ProgressBar1.Value = 0
  ProgressBar1.Visible = True

  Goal = Val(PwtOut.Text)
  Accuracy = 0.05
  Direction = 0        'first time thru step will be halved
  Step = 0.02          'before it is applied
  rho = Val(OutDensity.Text)
  rref = Val(OutBurnRate.Text)
  trefin = Val(InTemp.Text)
  trefout = Val(OutTemp.Text)
  PressOut(0) = 14.7

  nexp = Val(OutN.Text)
  CS = Val(CStar.Text) * 12
  BSModifier = 1
```

FIG. 6A

```
Trial = 0

Do
   Trial = Trial + 1
   ThisValue = 0
   For i = 1 To InPoints
      If i / Int(InPoints / 10) = Int(i / Int(InPoints / 10)) Then
         StatusText.Caption = "Generating Converted File... " & Chr$(13) & "Trial #"
         & Trial
         ProgressBar1.Value = i / InPoints * 100
         Refresh
      End If
      WBurnOut(i) = WBurnIn(i) * OutWebDepth.Text / InWebDepth.Text    'eq 10
      PercentWeb = WBurnOut(i) / OutWebDepth.Text
      BSurfOut(i) = BSurfIn(i) * BSModifier       'eq 11
      brate = rref * (PressOut(i - 1) / PREF) ^ nexp      'eq 12
      mpdot = brate * (BSurfOut(i) + BSurfOut(i - 1)) / 2 * rho 'eq 13
      If AtFOut.Text <> At0.Text Then
         AtOut(i) = (At0.Text + (Sqr(4 * AtIn(i) / PI) - At0.Text) / (AtF.Text - At0
         .Text) * (AtFOut.Text - At0.Text)) ^ 2 / 4 * PI 'eq 14
      Else
         AtOut(i) = AtIn(i)
      End If
      ' Values from form (At0.Text, etc) are actually diameters, not areas)
      PressOut(i) = (mpdot * CS) / (AtOut(i) * G)      'eq 15
      If PressOut(i) < 14.7 Then
         PressOut(i) = 14.7
      End If
      CfOut(i) = cfIn(i) * (1 + (ThrustCoefFactor.Text / 100) * (trefout - trefin)
      /   100)      'eq 16
      ThrustOut(i) = CfOut(i) * PressOut(i) * AtOut(i)      'eq 17
      TimeOut(i) = TimeOut(i - 1) + (WBurnOut(i) - WBurnOut(i - 1)) / brate    'eq
      18
      ThisValue = ThisValue + (mpdot + PrevMpdot) / 2 * (TimeOut(i) - TimeOut(i - 1
      ))      'eq(8)
      PrevMpdot = mpdot

      BBPGraph.Line (WBurnOut(i - 1), MaxBS * 1.3 - BSurfOut(i - 1))-(WBurnOut(i),
      MaxBS * 1.3 - BSurfOut(i)), BLUE
      ThrustGraph.Line (TimeOut(i - 1), MaxThrust * 1.3 - ThrustOut(i - 1))-
      (TimeOut(i), MaxThrust * 1.3 - ThrustOut(i)), BLUE

   Next i
   If Sgn(Goal - ThisValue) <> Sgn(Direction) Then
      Step = Step / 2
      Direction = Sgn(Goal - ThisValue)
   End If
   If Sgn(Goal - ThisValue) > 0 Then
      BSModifier = BSModifier + Step
   Else
      BSModifier = BSModifier - Step
   End If

Loop Until Abs(Goal - ThisValue) <= Accuracy

StatusLabel.Visible = False
StatusText.Visible = False
ProgressBar1.Visible = False
DataCutFrame.Visible = True
Refresh

CStar.Text = Format(CS / 12, "0")
```

FIG. 6B

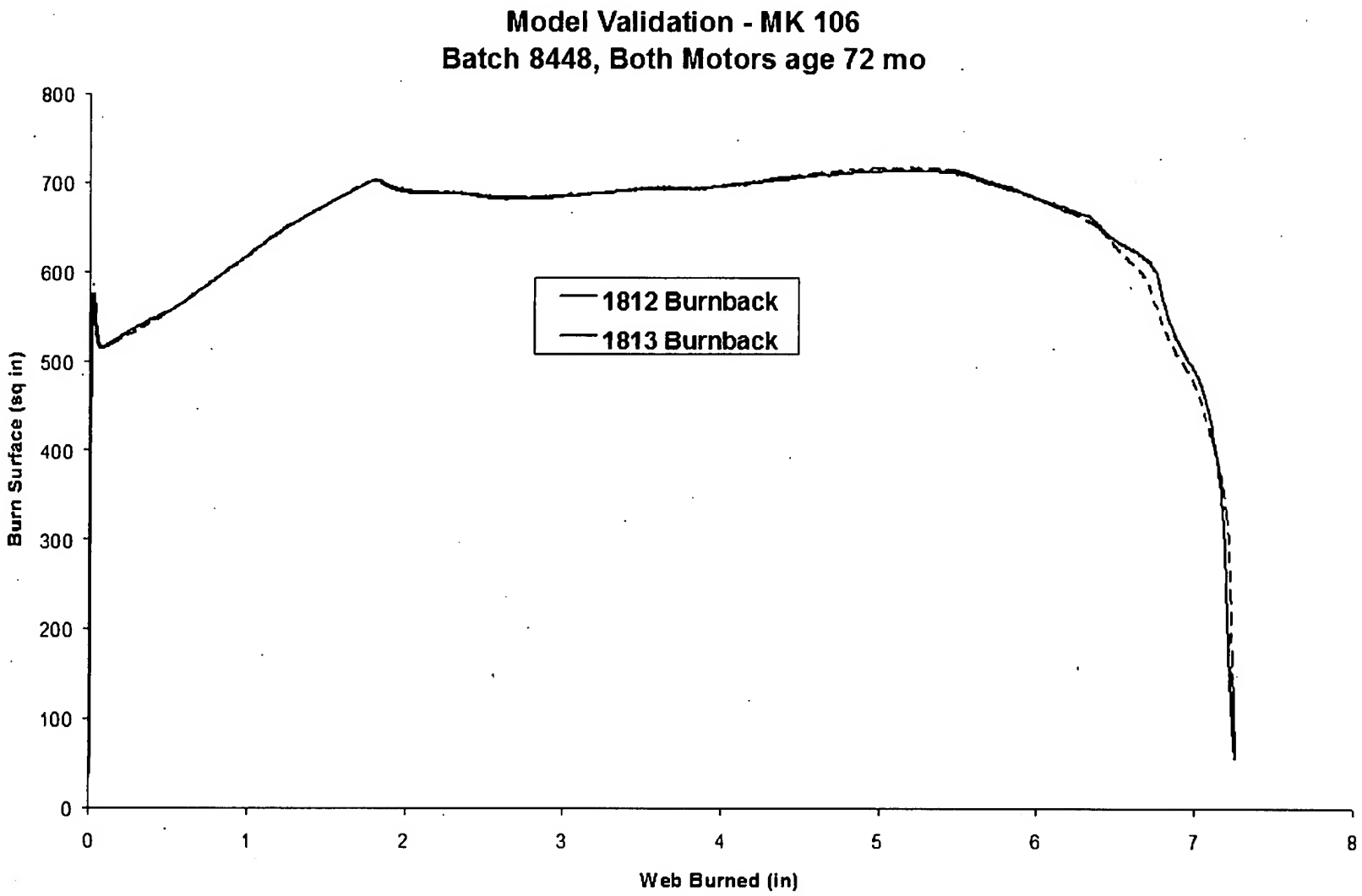
Model Validation - MK 106
Batch 8448, Both Motors age 72 mo
Burn Surface (sq in)
800
700
600
500
400
300
200
100
0
0
1
2
3
4
5
6
7
8
Web Burned (in)
1812 Burnback
1813 Burnback

FIG. 7

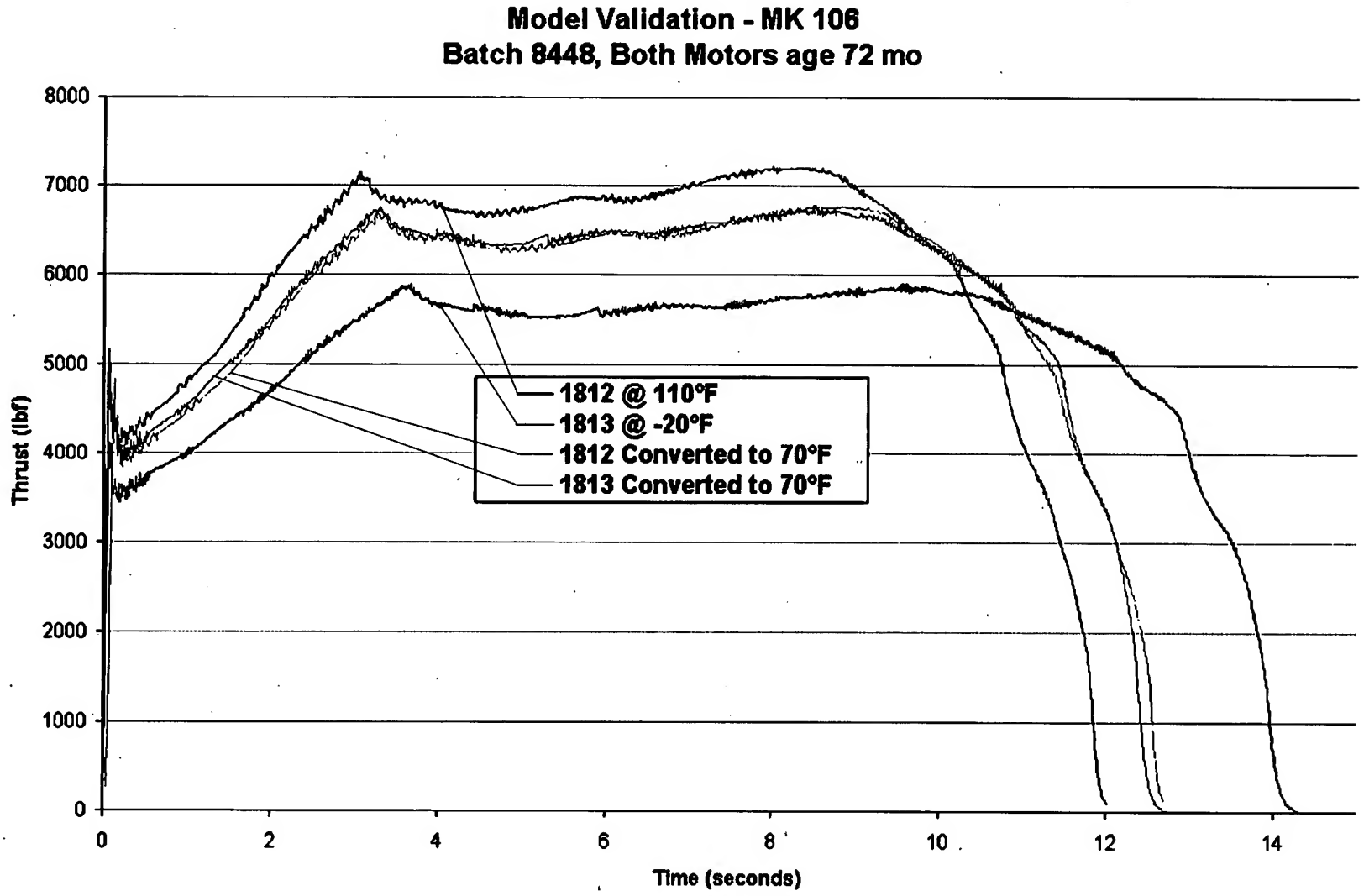
Model Validation - MK 106
Batch 8448, Both Motors age 72 mo
Thrust (lbf)
8000
7000
6000
5000
4000
3000
2000
1000
0
0
2
4
6
8
10
12
14
Time (seconds)
1812 @ 110°F
1813 @ -20°F
1812 Converted to 70°F
1813 Converted to 70°F

FIG. 8

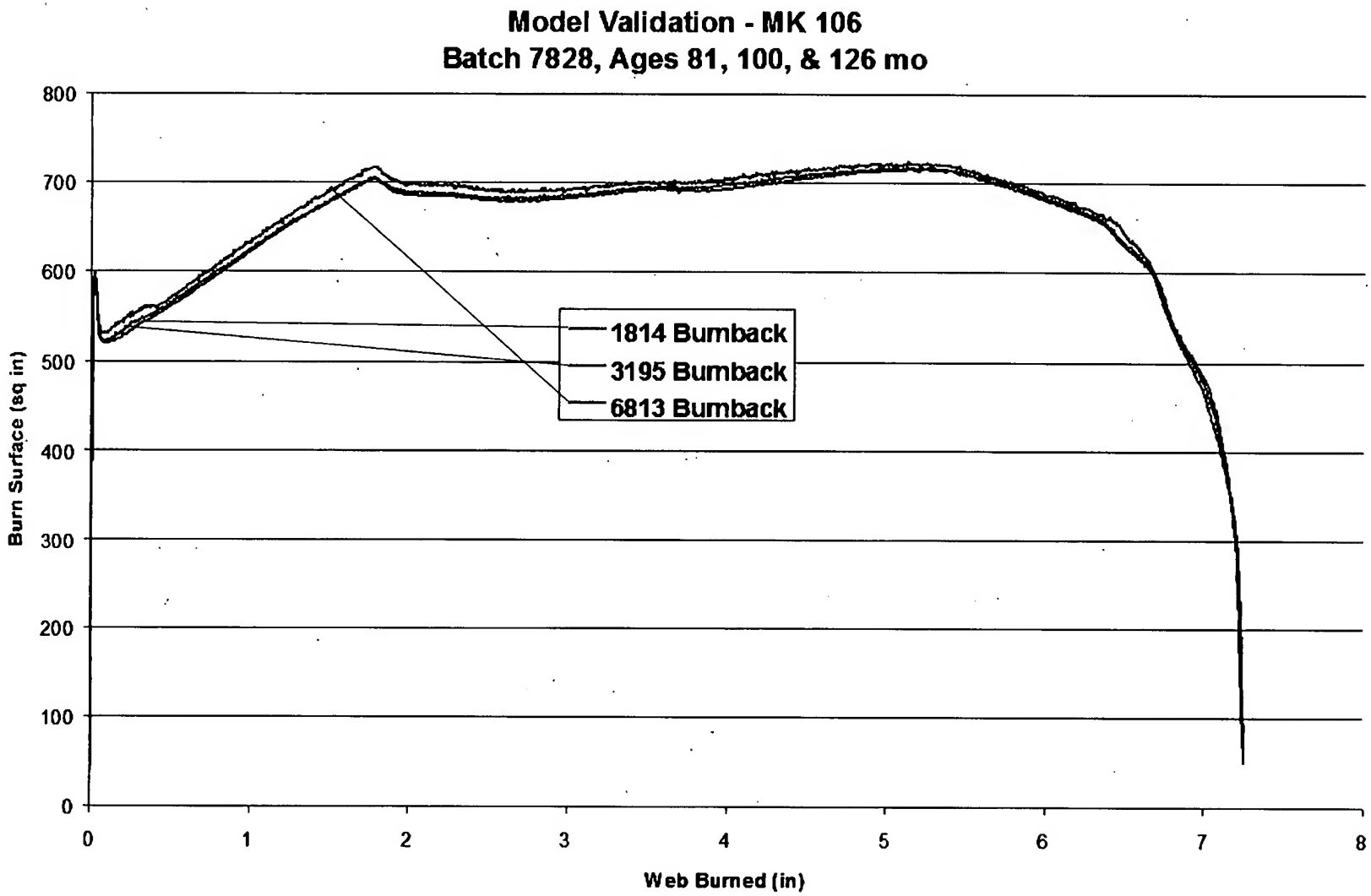
Model Validation - MK 106
Batch 7828, Ages 81, 100, & 126 mo
Burn Surface (sq in)
800
700
600
500
400
300
200
100
0
0
1
2
3
4
5
6
7
8
Web Burned (in)
1814 Bumback
3195 Bumback
6813 Bumback

FIG. 9

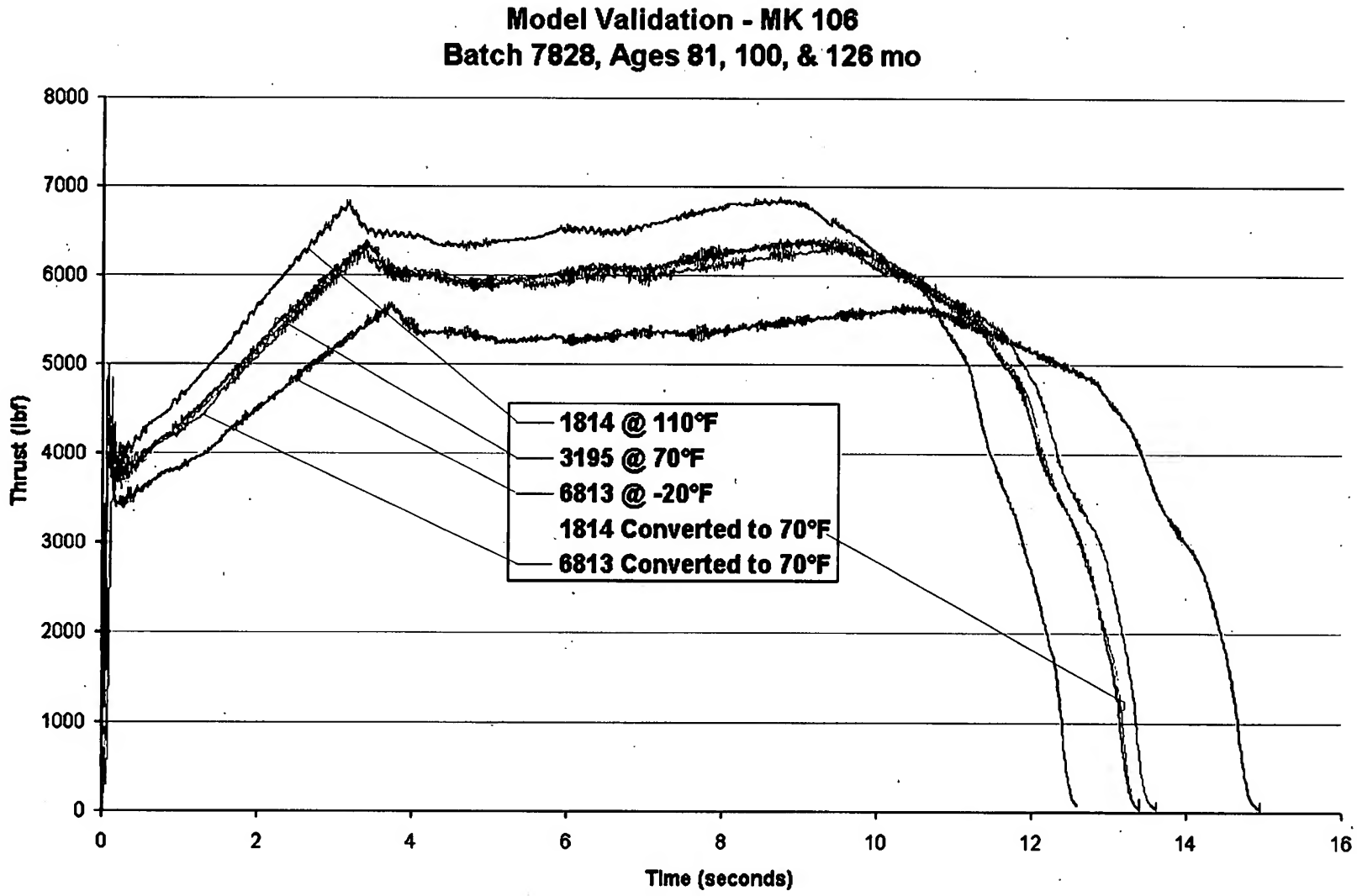
Model Validation - MK 106
Batch 7828, Ages 81, 100, & 126 mo
Thrust (lbf)
8000
7000
6000
5000
4000
3000
2000
1000
0
0
2
4
6
8
10
12
14
16
Time (seconds)
1814 @ 110°F
3195 @ 70°F
6813 @ -20°F
1814 Converted to 70°F
6813 Converted to 70°F

FIG. 10

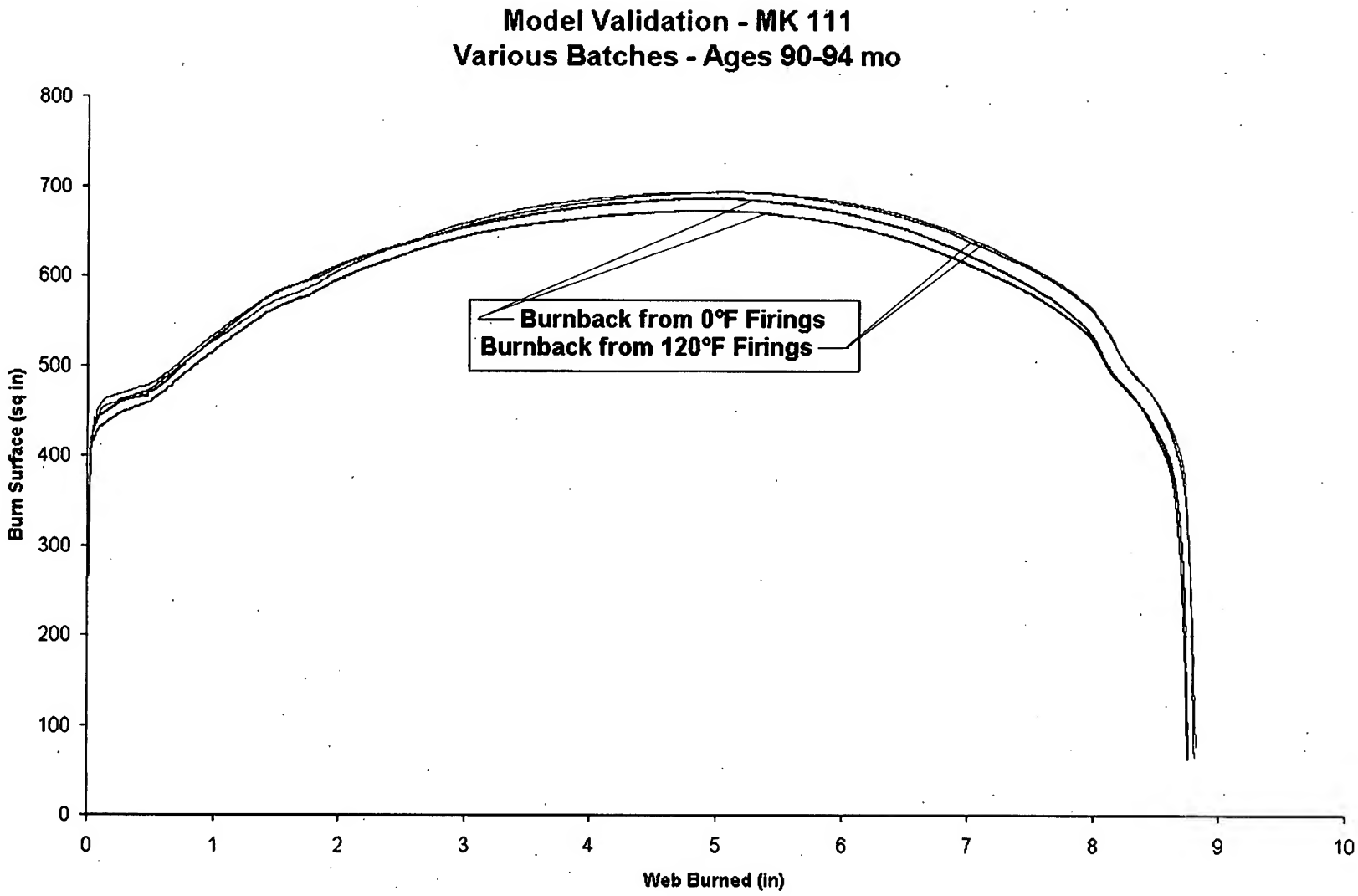
Model Validation - MK 111
Various Batches - Ages 90-94 mo
Burn Surface (sq in)
800
700
600
500
400
300
200
100
0
0
1
2
3
4
5
6
7
8
9
10
Web Burned (in)
Burnback from 0°F Firings
Burnback from 120°F Firings

FIG. 11

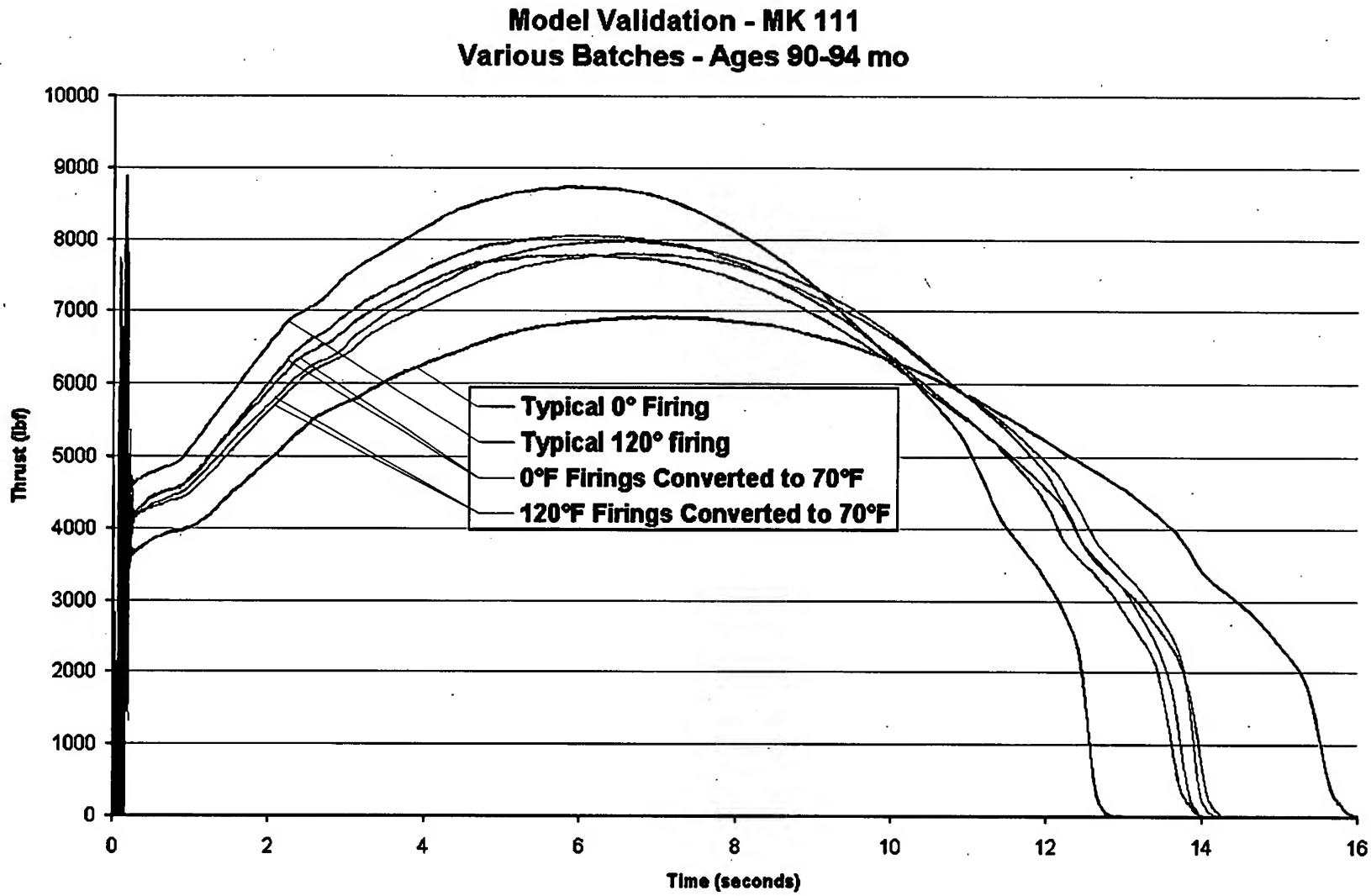
Model Validation - MK 111
Various Batches - Ages 90-94 mo
Thrust (lbf)
10000
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
0
2
4
6
8
10
12
14
16
Time (seconds)
Typical 0° Firing
Typical 120° firing
0°F Firings Converted to 70°F
120°F Firings Converted to 70°F

FIG. 12

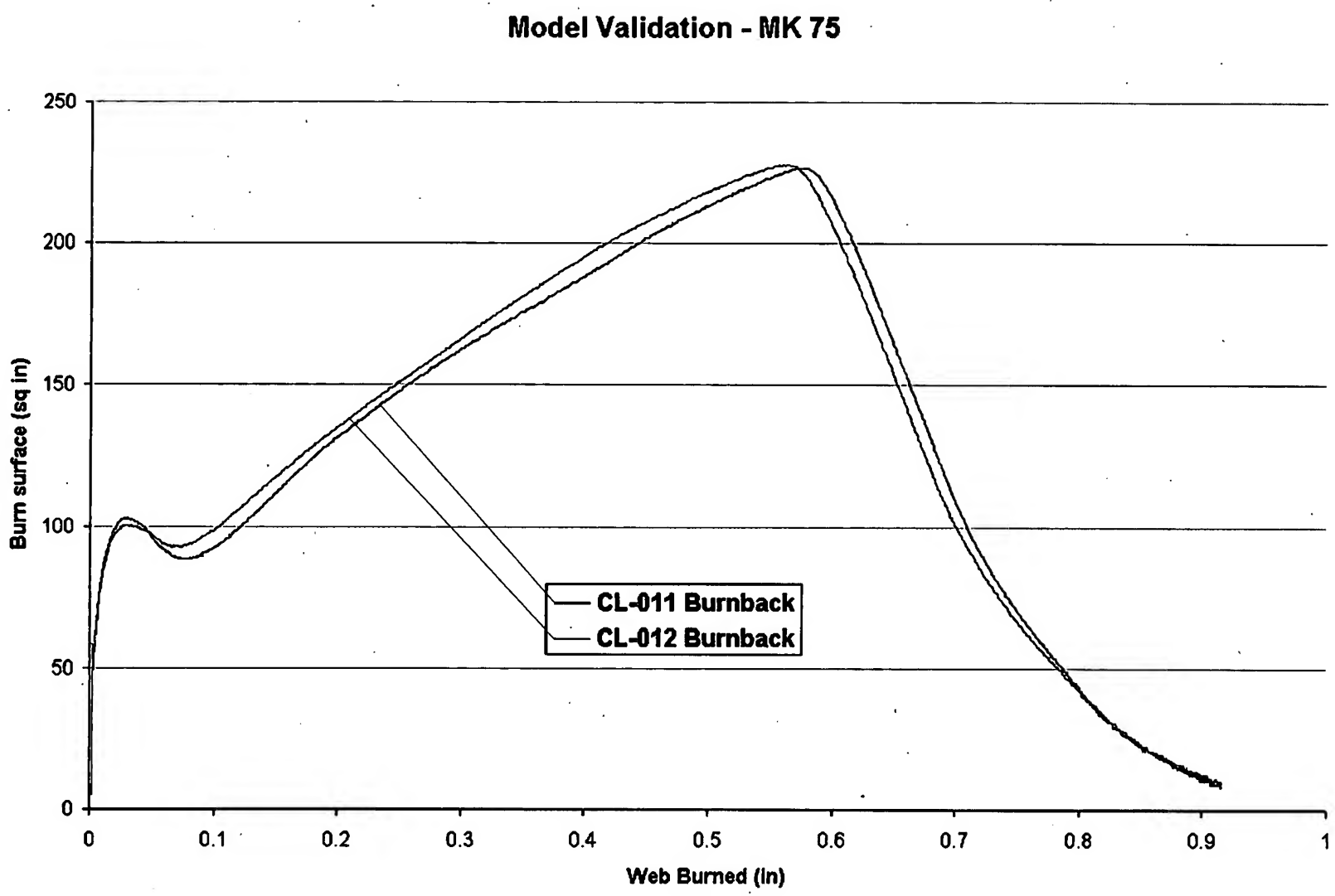
Model Validation - MK 75
Burn surface (sq in)
250
200
150
100
50
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Web Burned (in)
CL-011 Burnback
CL-012 Burnback

FIG. 13

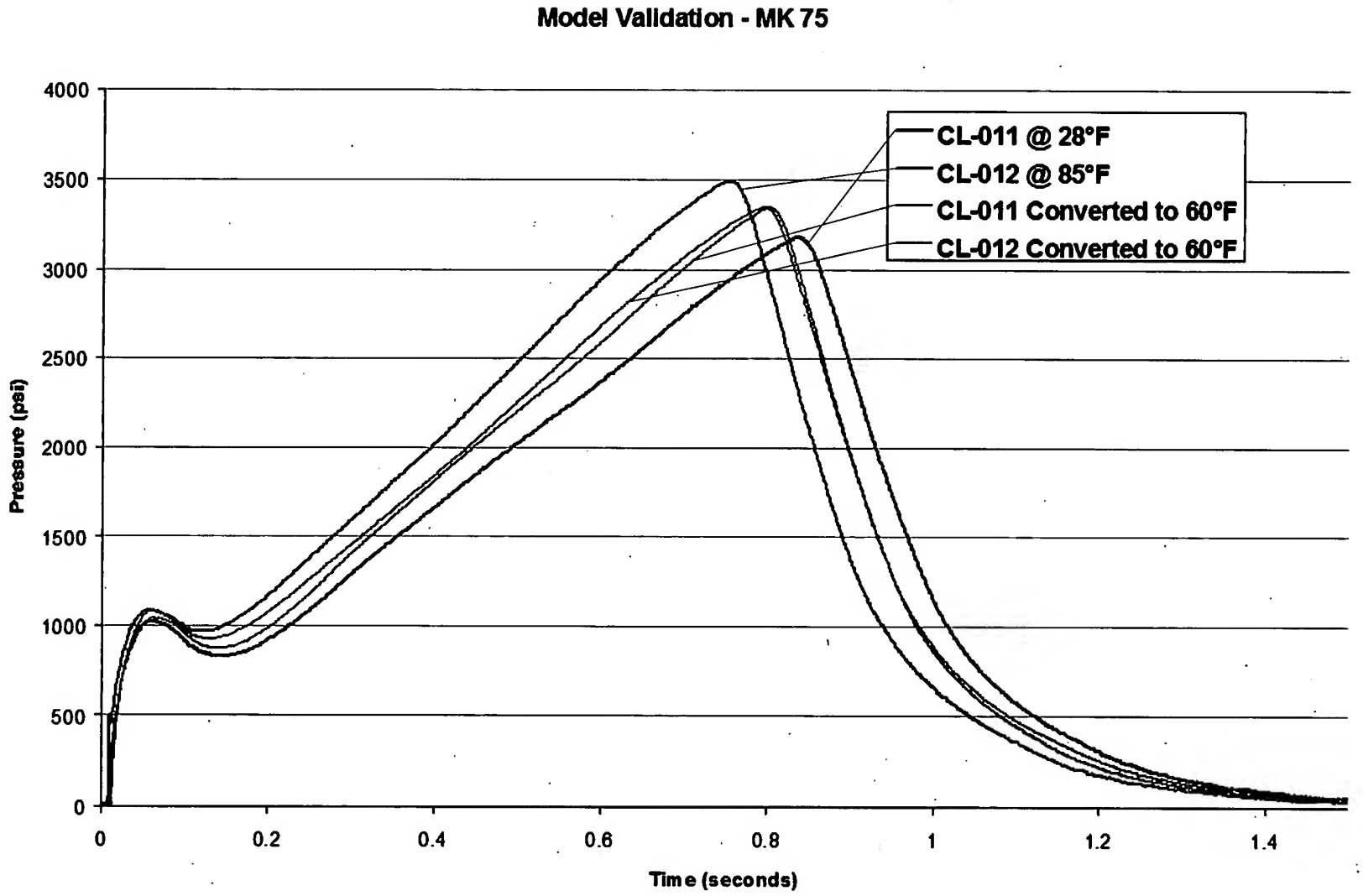
Model Validation - MK 75
4000
3500
3000
2500
2000
1500
1000
500
0
Pressure (psi)
0
0.2
0.4
0.6
0.8
1
1.2
1.4
Time (seconds)
CL-011 @ 28°F
CL-012 @ 85°F
CL-011 Converted to 60°F
CL-012 Converted to 60°F

FIG. 14